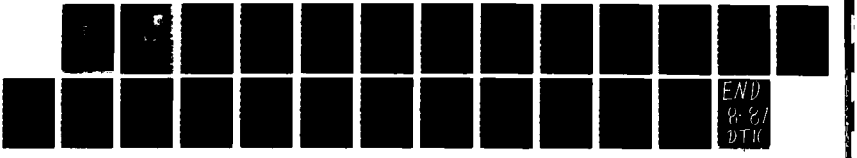
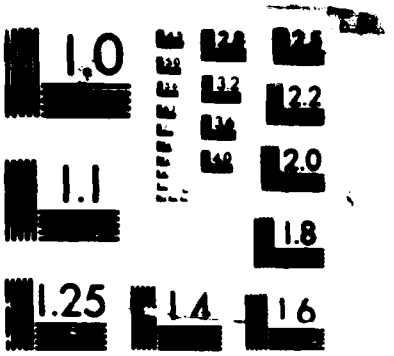
AD-A182 016 INTEGRATED INFORMATION SUPPORT SYSTEM (IISS) VOLUME 5 1/1
COMMON DATA MODEL S (U) GENERAL ELECTRIC CO
SCHENECTADY NY PRODUCTION RESOURCES CONSU
UNCLASSIFIED J L ALTHOFF 01 NOV 85 DS-620141330 F/G 12/5 NL





MICROCOPY RESOLUTION TEST CHART

AFWAL-TR-86-4006
Volume V
Part 30

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)
Volume V - Common Data Model Subsystem
Part 30 - File Utilities Development Specification

General Electric Company
Production Resources Consulting
One River Road
Schenectady, New York 12345

Final Report for Period 22 September 1980 - 31 July 1985

November 1985

Approved for public release; distribution is unlimited.

MATERIALS LABORATORY
AIR FORCE WRIGHT AERONAUTICAL LABORATORIES
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AFB, OH 45433-6533

NOTICE

When Government drawings specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS) At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

DAVID L. JUDSON, PROJECT MANAGER
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

5 Aug 1986
DATE

FOR THE COMMANDER:

GERALD C. SHUMAKER, BRANCH CHIEF
AFWAL/MLTC
WRIGHT PATTERSON AFB OH 45433

7 Aug 86
DATE

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/MLTC, W-PAFB, OH 45433 to help us maintain a current mailing list."

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

A182 016

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION: Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution is unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S): AFVAL-TR-86-4006 Vol V, Part 30
6a NAME OF PERFORMING ORGANIZATION: General Electric Company Production Resources Consulting; 6b OFFICE SYMBOL (If applicable)	7a NAME OF MONITORING ORGANIZATION: AFVAL/MLTC
6c ADDRESS (City, State and ZIP Code): 1 River Road Schenectady, NY 12345	7b ADDRESS (City, State and ZIP Code): WPAFB, OH 45433-6533
8a NAME OF FUNDING/SPONSORING ORGANIZATION: Materials Laboratory Air Force Systems Command, USAF; 8b OFFICE SYMBOL (If applicable): AFVAL/MLTC	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER: F33615-80-C-5155
8c ADDRESS (City, State and ZIP Code): Wright-Patterson AFB, Ohio 45433	10 SOURCE OF FUNDING NOS: PROGRAM ELEMENT NO. 78011F; PROJECT NO. 7500; TASK NO. 62; WORK UNIT NO. 01
11 TITLE (Include Security Classification): (See Reverse)	
12 PERSONAL AUTHOR(S): Althoff, J. L.	
13a TYPE OF REPORT: Final Technical Report; 13b TIME COVERED: 22 Sept 1980 - 31 July 1985; 14 DATE OF REPORT (Yr., Mo., Day): 1985 November; 15 PAGE COUNT: 21	
16 SUPPLEMENTARY NOTATION: ICAM Project Priority 6201	The computer software contained herein are theoretical and/or references that in no way reflect Air Force-owned or -developed computer software.

17 COSATI CODES		
FIELD	GROUP	SUB GR
1308	0905	

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This document is the development specification establishing the functional requirements of the IISS Configuration Item File Utilities which provide file transfer, file delete and unique file naming services to other conponents of IISS.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT: UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21 ABSTRACT SECURITY CLASSIFICATION: Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL: David L. Judson	22b TELEPHONE NUMBER (Include Area Code): 513-255-6076; 22c OFFICE SYMBOL: AFVAL/MLTC

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

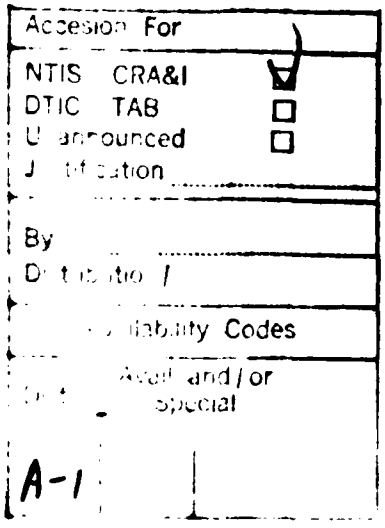
Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

11. Title

Integrated Information Support System (IISS)
Vol V - Common Data Model Subsystem
Part 30 - File Utilities Development Specification

A S D 86 1435
17 Jul 1986



PREFACE

This development specification covers the work performed under Air Force Contract F33615-80-C-5155 (ICAM Project 6201). This contract is sponsored by the Materials Laboratory, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio. It was administered under the technical direction of Mr. Gerald C. Shumaker, ICAM Program Manager, Manufacturing Technology Division, through Project Manager, Mr. David Judson. The Prime Contractor was Production Resources Consulting of the General Electric Company, Schenectady, New York, under the direction of Mr. Alan Rubenstein. The General Electric Project Manager was Mr. Myron Hurlbut of Industrial Automation Systems Department, Albany, New York.

Certain work aimed at improving Test Bed Technology has been performed by other contracts with Project 6201 performing integrating functions. This work consisted of enhancements to Test Bed software and establishment and operation of Test Bed hardware and communications for developers and other users. Documentation relating to the Test Bed from all of these contractors and projects have been integrated under Project 6201 for publication and treatment as an integrated set of documents. The particular contributors to each document are noted on the Report Documentation Page (DD1473). A listing and description of the entire project documentation system and how they are related is contained in document FTR620100001, Project Overview.

The subcontractors and their contributing activities were as follows:

TASK 4.2

Subcontractors	Role
Boeing Military Aircraft Company (BMAC)	Reviewer.
D. Appleton Company (DACOM)	Responsible for IDEF support, state-of-the-art literature search.
General Dynamics/ Ft. Worth	Responsible for factory view function and information models.

Subcontractors	Role
Illinois Institute of Technology	Responsible for factory view function research (IITRI) and information models of small and medium-size business.
North American Rockwell	Reviewer.
Northrop Corporation	Responsible for factory view function and information models.
Pritsker and Associates	Responsible for IDEF2 support.
SofTech	Responsible for IDEF0 support.

TASKS 4.3 - 4.9 (TEST BED)

Subcontractors	Role
Boeing Military Aircraft Company (BMAC)	Responsible for consultation on applications of the technology and on IBM computer technology.
Computer Technology Associates (CTA)	Assisted in the areas of communications systems, system design and integration methodology, and design of the Network Transaction Manager.
Control Data Corporation (CDC)	Responsible for the Common Data Model (CDM) implementation and part of the CDM design (shared with DACOM).
D. Appleton Company (DACOM)	Responsible for the overall CDM Subsystem design integration and test plan, as well as part of the design of the CDM (shared with CDC). DACOM also developed the Integration Methodology and did the schema mappings for the Application Subsystems.

Subcontractors	Role
Digital Equipment Corporation (DEC)	Consulting and support of the performance testing and on DEC software and computer systems operation.
McDonnell Douglas Automation Company (McAuto)	Responsible for the support and enhancements to the Network Transaction Manager Subsystem during 1984/1985 period.
On-Line Software International (OSI)	Responsible for programming the Communications Subsystem on the IBM and for consulting on the IBM.
Rath and Strong Systems Products (RSSP) (In 1985 became McCormack & Dodge)	Responsible for assistance in the implementation and use of the MRP II package (PIOS) that they supplied.
SofTech, Inc.	Responsible for the design and implementation of the Network Transaction Manager (NTM) in 1981/1984 period.
Software Performance Engineering (SPE)	Responsible for directing the work on performance evaluation and analysis.
Structural Dynamics Research Corporation (SDRC)	Responsible for the User Interface and Virtual Terminal Interface Subsystems.

Other prime contractors under other projects who have contributed to Test Bed Technology, their contributing activities and responsible projects are as follows:

Contractors	ICAM Project	Contributing Activities
Boeing Military Aircraft Company (BMAC)	1701, 2201, 2202	Enhancements for IBM node use. Technology Transfer to Integrated Sheet Metal Center (ISMC).

Contractors	ICAM Project	Contributing Activities
Control Data Corporation (CDC)	1502, 1701	IISS enhancements to Common Data Model Processor (CDMP).
D. Appleton Company (DACOM)	1502	IISS enhancements to Integration Methodology.
General Electric	1502	Operation of the Test Bed and communications equipment.
Hughes Aircraft Company (HAC)	1701	Test Bed enhancements.
Structural Dynamics Research Corporation (SDRC)	1502, 1701, 1703	IISS enhancements to User Interface/Virtual Terminal Interface (UI/VTI).
Systran	1502	Test Bed enhancements. Operation of Test Bed.

TABLE OF CONTENTS

SECTION 1

SCOPE

1.1 Identification

This specification establishes the performance, development, test, and qualification requirements of a set of computer programs identified as Configuration Item File Utilities.

This CI constitutes one of the subsystems of the Common Data Model Processor (CDMP) which is described in the System Design Specification (SDS) for the ICAM Integrated Support System (IISS). The CDMP scope is based on a logical concept of subsystem modules that interface with other external systems of the IISS. The CDMP has been portrayed with three configuration items: The Precompiler, the Distributed Request Supervisor (DRS), and the Aggregator. In addition, File Transfer File Delete and File Namer are included as utilities to be used by the CDMP and other IISS components. The following narrative portrays the scope of the CDMP in a compile and a runtime environment.

Operation of CDMP at Precompiler Time

To make use of the CDM, a user embeds neutral data manipulation language (NDML) commands in COBOL programs. The Precompiler scans COBOL application program source code, identifies and parses NDML commands, applies transformations from external schema form to conceptual schema form, identifies data location, decomposes the commands to appropriate single database requests, applies transformations from conceptual schema form to internal schema form, and selects appropriate access paths through identified databases.

The Precompiler generates code that will be activated at run time to access the identified databases and to perform required internal-to-conceptual-to-external transforms. It also generates query graphs that control the management of the run-time transaction processing by the DRS.

Operation of CDMP at Application Run Time

The DRS selects the appropriate sequence in which to process multiple single database accesses that may result from

one NDML command. The scheduling of the accesses is done at run time based upon query graph information and a count of each set of qualifying records. The DRS controls the distributed activities presenting the aggregated result to the requesting application program.

The DRS invokes the execution of request processors which may exist at multiple workstations. Once activated, a request processor accesses the target data base and outputs qualifying records on a sequential output file. The request processor signals the DRS when it has completed processing a request with the name of the file that contains the qualifying records along with a volume count.

The DRS analyzes volume count from the various query processing operations in order to minimize file transportation costs. Based on the cost data, it activates the File Transfer utility to transport the least cost files to the appropriate host for aggregation processing.

The Aggregator responds to control messages from the DRS. It receives intermediate sets of qualifying records on sequential files resulting from single database accesses. It joins them appropriately to prepare the aggregate results.

After the final aggregation has been accomplished, the Stager/Scheduler initiates the conceptual to external transformer to convert the final data into the form expected by the original request in the application.

Any intermediate files that were created during the run time operation of the CDMP are deleted by requests to the File Delete Utility.

1.2 Functional Summary

Although the File Transfer utility will be usable by test bed components that require a basic file transfer capability, the primary functions are:

- To transfer the results of a request processor retrieval to another host where it will be aggregated with data from another request processor.
- To transfer the results of an intermediate aggregation step to another host where is will be further aggregated with other data.

- To transfer the final aggregated data to the [illegible] conceptual external transformer where it will be converted to external format and presented back to the original requestor

- To transfer generated programs from one [illegible] another

The primary uses of the File Delete are

- To delete a file containing request processor results after it has been sent to another host to be aggregated with associated query processor results

- To delete a file containing intermediate aggregation results

- To delete a file containing the final aggregated results after it has been transformed from Conceptual to External format by the C/E transformer

- To delete a file containing the final results from an NDML request after the information has been processed by the requesting application.

- To delete temporary files used by the NDML precompiler.

The primary uses of the File Namer are:

- To provide system wide unique file names for any requesting process.

- Specifically, unique file names are needed for temporary query results files.

- Specifically, unique file names are needed by the NDML precompiler for temporary files and for storage of generated programs.

SECTION 2

DOCUMENTS

2.1 Reference Documents

1. DeJean, J.P., Test Bed System Development Specification, General Electric Company, Schenectady, New York, November 9, 1982.

2.2 Terms and Abbreviations

Attribute Use Class: (AUC)

Conceptual Schema: (CS)

Common Data Model Processor: (CDMP)

Common Data Model: (CDM) Describes common data application process formats, form definitions, etc, of the IISS and includes conceptual schema, external, internal schemas, and schema transformation operators.

Data Field: (DF) An element of data in the external schema. It is by this name that an NDML programmer reference data.

Database Management System: (DBMS)

Distributed Request Supervisor: (DRS) This IISS CDM subsystem configuration item controls the execution of distributed NDML queries and non distributed updates.

Domain: A logical definition of legal attribute class values.

Domain Constraint: Predicate that applies to a single domain.

External Schema: (ES)

Forms: Structured views which may be imposed on windows or other forms. A form is composed of fields where each field is a form, item, or window.

Forms Processor: (FP) A set of callable execution time

routines available to an application program for form processing.

Internal Schema: (IS)

Integrated Information Support System: (IISS) A test computing environment used to investigate, demonstrate and test the concepts of information management and information integration in the context of Aerospace Manufacturing. The IISS addresses the problems of integration of data resident on heterogeneous databases supported by heterogeneous computers interconnected via a local Area Network.

Mapping: The correspondence of independent objects in two schemas: ES to CS or CS to IS.

Network Transaction Manager: (NTM) Performs the coordination, communication and housekeeping functions required to integrate the application processes and system services resident on the various hosts into a cohesive system.

Neutral Data Manipulation Language: (NDML) A language developed by the IISS project to provide uniform access to common data, regardless of database manager or distribution criteria. It provides distributed retrieved and single node updates.

ORACLE: Relational DBMS based on the SQL (Structured Query Language, a product of ORACLE Corp, Menlo Park, CA). The CDM is an ORACLE database.

Parcel: A sequential file containing sections source code of the input application program.

Request Processor: (RP) A COBOL program that will satisfy a retrieval or update NDML subtransaction against a particular Database Management System.

User Interface: (UI) Controls the user's terminal and interfaces with the rest of the system.

Virtual Terminal Interface: (VTI) Performs the interfacing between different terminals and the UI. This is done by defining a specific set of terminal features and protocols which must be supported by UI software which constitutes the Virtual Terminal Definition. Specific terminals are then mapped against the Virtual Terminal software by specific software modules

written for each type of real terminal supported.

SECTION 3

REQUIREMENTS

3.1 Computer Program Definition

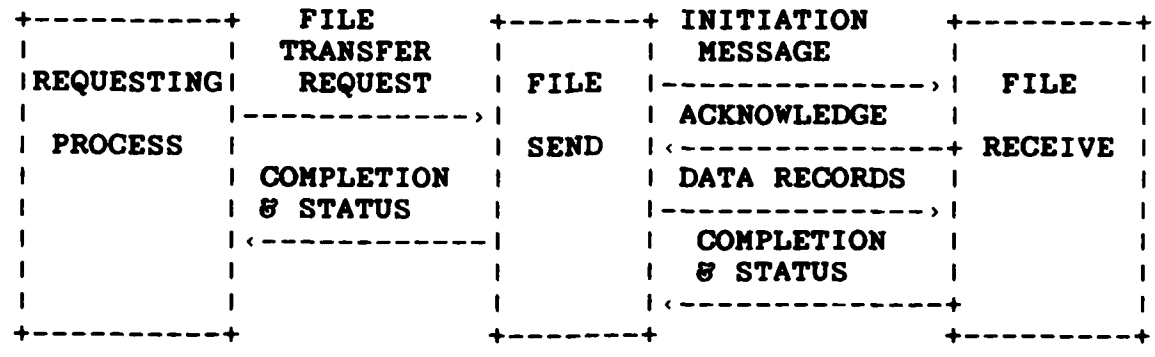
3.1.1 System Capacities

The File Transfer operates on ASCII files with records that are fixed in length (all same size) or variable length record files that do not exceed a maximum records exceeding the maximum are truncated.

The File Delete operates on ASCII file.

3.1.2 Interface Requirements

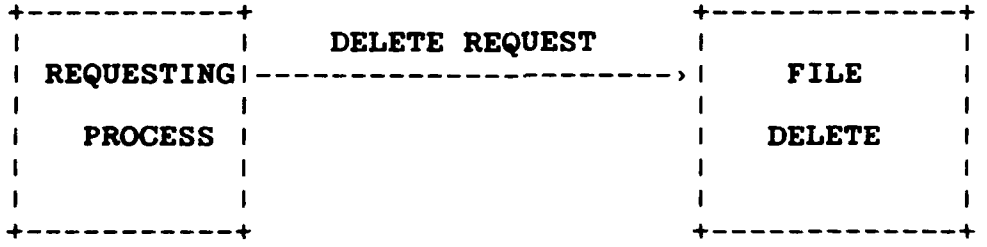
3.1.2.1 Interface Block Diagram



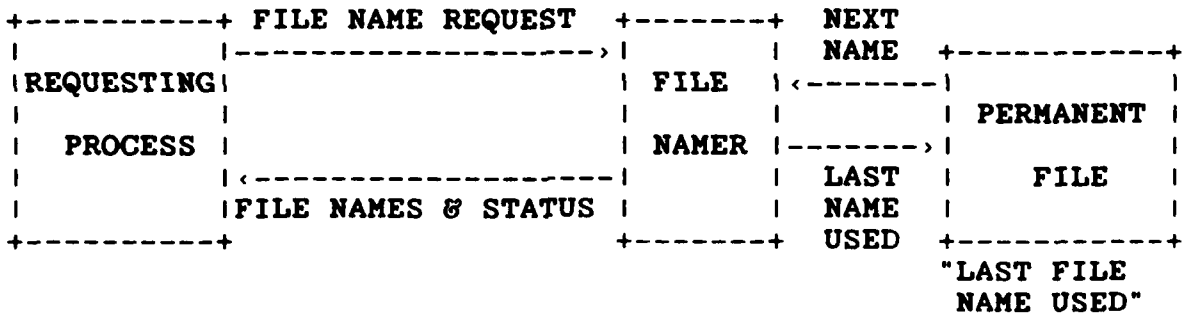
The File Transfer CI is composed of two functional components, File Send and File Receive.

A requesting process sends a File Transfer message to the File Send at the host where the original file to be transmitted resides. The File Send then sends an initiation message to the File Receive at the host where the file is to be created. After receiving acknowledgment from File Receive the File Send reads the source file and creates messages containing the file data.

It transmits these messages to the File Receive where they are reconstructed into a new file. When the file transfer is completed, File Send notifies the requesting process.



A requesting process sends a File Delete request in the form of a message to the host where the file to be deleted resides.



A requesting process sends File Name request in the form of a message to the File Name queue server. After generating the next group of names, the File Namer sends a reply message with a status to the requesting process. The file "LAST FILE NAME USED" is used to periodically record the last file name assigned for each host.

3.2 Detailed Functional Requirements

The following subsections document the File Utility's major functions and describes the input-output requirement.

3.2.1 Function FSD - File Send

The file send function transfers files from host to host in

the testbed environment. The file send function residing at one host sends it contents to a file receive function on a remote host. The file send receives file characteristics and operating instructions from a requesting process. It in turn passes control information to the file receive function.

3.2.1.1 Inputs (From requesting process)

A. Name of file to be sent.
B. Record length of records contained in file.
C. Name of file to be built.
D. Logical location of where file is to be sent.
E. Indicator on whether to delete or keep input file after transfer.
F. "Ready to Receive" (from File Receive).
G. "End of File Transfer" or "File Transfer Unsuccessful" (from File Receive).

3.2.1.2 Processing

A. Receive File Transfer initiation message from requesting process via the NTM, to Transfer File. The message contains the inputs A thru E message from 3.2.1.1, above.
B. Send message to location where file is to be sent to initiate file receive routine. Send name of file to be created and record length, from step B, above.
C. Receive "ready to receive" message from file receive routine.
D. Open input file to be sent.
E. Read input records and place in output buffer. Continue until output buffer is full or all records from file have been read. Keep a count of total number of records in file.
F. Send message to file receive routine containing data from E, above.
G. Repeat E and F, as necessary, to send entire file.
H. Send message to file receive routine inidcating all records have been sent and total record count from step E.
I. Wait for "End of File Transfer" or "File Transfer Unsuccessful" completion message from File Receive.

    1. If "File Transfer Unsuccessful" is received, check to see if this is first or second attempt to send the file. If first, try again by restarting from the beginning. If second try, send unsuccessful

message to requesting process and terminate.

2. If "End of File Transfer" continue with K, below.

K. Check input parameter E, under 3.2.1.1 above and delete original input file if the delete parameter is set. Note: use the File Delete utility to execute this step.

L. Send a completion message to the process requesting the file transfer.

M. Terminate the File Send.

3.2.1.3 Outputs

A. Message to receiving host to initiate File Receive.

- Name of file to be created
- Record length

B. Records containing data to be transferred.

C. "End of File" message (containing count of number of records sent).

D. Completion message to requesting process containing transfer status.

3.2.2 Function FRC - File Receive

The File Receive function operates in conjunction with the File Send function in the test bed environment. File Send, residing at one host, sends its contents to File Receive on a remote host. Prior to receiving any of the file contents, File Receive receives control information from File Send that contains the name of the file to be built and other file characteristics.

3.2.2.1 Inputs

A. Message from File Send to initiate receive process.

- Name of file to be created.
- Record length of records in the file.

B. Data records to build new file.

C. "End of File" message from File Send, which includes a count of the records sent.

3.2.2.2 Processing

A. Receive initiation message from File Send. (See input A in 3.2.1.1, above).

B. Create (open) output file to be written.

C. Send "Ready to Receive" message to File Send.

D. Read input records being sent by File Send and place on output file opened in B, above. Keep a count of number of logical records received and written.

E. Repeat Step D until all records have been written to output file as denoted by "End of File" message from File Send.

F. Compare count of number of logical records written against number of records sent. Then take one of the following actions:

- If the counts are equal, send "End of File transfer" message to File Receive.
- If the counts are not equal, delete the output file being created, and send "File Transfer Unsuccessful" message to the File Send routine, terminate.

G. Close the output file.

H. Terminate.

3.2.2.3 Outputs

A. "Ready to Receive" message to File Send.

B. Newly created data file.

C. "End of File Transfer" message or "File Transfer Unsuccessful" message

3.2.3 Function FDL - File Delete

The File Delete utility purges files residing on the local host. The request to delete a file can originate on any host in the IISS environment.

3.2.3.1 Input

A. File Delete initiation message containing name of file to be deleted.

3.2.3.2 Processing

A. An interface module, upon receiving the file name and its host of residence will issue an operating system call to delete or purge the file if the file is on the same host as the interface module.
B. If the file is "off-host", a message is sent to the file delete process on that host which will then execute a call to the operating system (on local host) to delete the file using the file name supplied in input A, above.
C. No status reply is expected from the file delete interface or the file delete process.

3.2.4 Function FNM - File Namer

The File Namer utility generates unique file names for any host node of the IISS. The request for file names can originate on any host in the IISS environment.

3.2.4.1 Input

A. File Namer must be told by the requester that it needs a file name. The fact that a call is made to the interface module indicates this.

B. The interface module may send a message to the file name queue server for a group of file names. The fact that a requesting message arrives at the queue server indicates that a group of file names is desired.

3.2.4.2 Processing

A. The file name interface maintains a group of file names recieved from the queue server. When this group has been passed out, one per request, the interface module

will request another group from the queue server.

B. The file name queue server will maintain an in-memory table of the last name assigned for each host. This was loaded from the "LAST FILE NAME USED" file at process initiation. When a request comes in, a block of names is generated by incrementing the numeric portion of the name. If the numeric portion "rolls" past all 9's, then the letter portion of the name is incremented. The group of file names is sent back to the requestor in the reply message.

C. After every 50th request, the current version of the next file name to use is written to the "LAST FILE NAME USED" file. This also is done when the NTM signals the system shutdown.

D. This function assumes no other users of the host computer will be generating file names of the type developed, as no operating system calls or checks are made. This can be insured by reserving a special directory or file name template for IISS use.

3.2.4.3 Output

A. A file image ("LAST FILE USED") of the next file names to be used is stored on disk to prevent re-use of file names across system shutdown and startup.

B. A group of file names is sent in a reply message to the requesting file name interface routine.

C. A single file name is returned from the file name interface routine to the requesting application.

SECTION 4

QUALITY ASSURANCE PROVISIONS

4.1 Introduction and Definitions

"Testing" is a systematic process that may be preplanned and explicitly stated. Test techniques and procedures may be defined in advancce and a sequence of test steps may be specified. "Debugging" is the process of isolation and correction of the cause of and error.

4.2 Computer Programming Test and Evaluation

The quality assurance provisions for test will consist of the normal testing techniques that are accomplished during the construction process. They consist of design and code walk-throughs, unit testing, and integration testing. These tests will be performed by the design team.

The integation test developed for the file utilities will consist of a number of test cases developed for other components of the CDM. Because file utilities are essentially service modules, they will be tested by other CDM components.

Unit testing will primarily involve testing each of the utilities with skeleton routines calling the "service" interface module directly.

SECTION 5

PREPARATION FOR DELIVERY

The implementation site for the constructed software will be the ICAM Integrated Support System (IISS) Test Bed Site located in Albany, New York. The software associated with the file utilities will be clearly identified and will include instructions on procedures to be followed for installation of the release.

U.S.Government Printing Office 1987 748 061 60912

END

8-87

DTIC